UNITED STATES PATENT AND TRADEMARK OFFICE

am

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/939,378	08/24/2001	Joseph Franklin Garvey	RAL920000124US1	3898

45503 7590 06/06/2005

DILLON & YUDELL LLP
8911 N. CAPITAL OF TEXAS HWY.,
SUITE 2110
AUSTIN, TX 78759

EXAMINER
VU, TUAN A

ART UNIT	PAPER NUMBER
2193	

DATE MAILED: 06/06/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/939,378	GARVEY, JOSEPH FRANKLIN	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE _3_ MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on _18 January 2005_.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) _1-8_ is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) _1-8_ is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

[illegible]

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 1/18/2005.

As indicated in Applicant's response, claims 1, 7-8 have been amended. Claims 1-8 are pending in the office action.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

3. Claims 1, and 4-5 are rejected under 35 U.S.C. 102(b) as being anticipated by Leeper et al., "Structured Assembly Language in VAX-11 MACRO", Feb. 1986, Proceedings of the 17th SIGCSE technical symposium on Computer Science education, Vol. 18,m issue 1(hereinafter Leeper).

As per claim 1, Leeper discloses an assembler for processing structured assembly language expressions, said assembler comprising:

program code means for recognizing a structured assembly language expression's mnemonics containing elements **argl cc arg2** (e.g. *algorithm* – pg. 54; IF *X* > *Y* -pg. 54; IF *X* >= *0* pg. 55; WHILE *NAME* <> *TRAILER* – pg. 57), wherein said **cc** is a condition code (> ; >=, <> -- Note: greater than, greater or equal to, not equal to are condition codes), wherein the form of said expression's mnemonics or the nature of one or more of said expression's elements selects a corresponding comparison opcode (e.g. CMPC3 – pg. 54; CMPL – pg. 55; CMPC3 – pg. 57 – Note: the form of the template expression symbols dictates a corresponding opcode), wherein said argl and said arg2 are valid arguments for said selected comparison opcode (Note: X and Y are inherently valid and type equivalent in order for opcode to function);

program code means for constructing a data structure referencing said argl, said arg2, said cc, and a branch destination (e.g. *constructs for IF-THEN, templates* – pg. 54, 3rd para);

program code means for generating a comparison opcode in response to elements of said data structure (e.g. (structured template) *CMPx* – pg. 54, 3rd para; *CMPx* – pg. 55, 1st para; *CMPx* – pg. 56, 6th para -- > (assembly language) CMPC3 – pg. 54; CMPL – pg. 55; CMPC3 – pg. 57);

program code means for generating a conditional branch based on said condition code in said data structure (e.g. *BGTR* – pg. 54, last para; *BGEQ, MNEGB* – pg. 55, 3rd para; *BNEQ, BEQL* – pg. 57, 2nd para);

program code means for generating a first branch location for execution to proceed as if said structured assembly language expression is true (e.g. *BGTR* – pg. 54, last para; *BEQ* – pg. 57, 2nd para);

program code means for generating a second branch location for execution to proceed as if said structured assembly language expression is false (e.g. *BNEQ* - pg. 57, 2nd para); and

program code means for generating a third branch location for execution to proceed to the end of said structured assembly language expression (e.g. *BEQL END_WHILE04* – pg. 57, 2nd para); and

program code means for indicating said branch destination (e.g. *WHILE04, END_WHILE04* – pg. 57, 2nd para) in said data structure is a branch to said first, said second, or said third branch locations.

As per claim 4, Leeper discloses means for not generating a comparison opcode in response to said data structure (e.g. THEN_BEGINnn: NOP ... ELSE_BEGINnn: ... pg. 55, top para)

As per claim 5, Leeper discloses assembling code generation by iterating over a vector of structured assembly language (SAL) structures of various forms (e.g. FOR LOOP CONSTRUCT, the WHILE LOOP CONSTRUCT, REPEAT-UNTIL CONSTRUCT – pg. 55-58 – Note: the combination of more than one logical expressions to make compounded logical expressions is implicitly disclosed in every programming language with complex iteration and compounded arguments type logical operations, hence building assembly language from combination of vectors like SAL templates as taught by Leeper is disclosed via putting together complex iteration statements and comparing compounded expressions).

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 2-3, and 6-8 are rejected under 35 U.S.C. 103(a) as being unpatentable over Leeper et al., "Structured Assembly Language in VAX-11 MACRO", Feb. 1986, Proceedings of the 17th SIGCSE technical symposium on Computer Science education, Vol. 18,m issue 1,pp. 53-60; in view of Curzon, Paul, "A Verified Compiler for a Structured Assembly Language", 1992, *International Workshop on Higher Order Logic Theorem Proving and Its Applications*, pp. 253-262(hereinafter Curzon) .

As per claim 2, Leeper discloses assembler further includes program code means for recognizing a structured assembly language expression's mnemonics having a format of condition code (re claim 1; *X* > *Y* -pg. 54; IF *X* >= *0* pg. 55) but does not explicitly disclose a **cc** form, wherein said cc is a condition code. Abstracting a operator into a more symbolic form ((e.g. <var1> <op> <var2>) was a known concept in high-level language at the time the invention was made and structured assembly language as taught by Leeper is such a form of high-level language with respect to assembly code. Hence, abstracting the condition operator or symbol as suggested by Leeper into a more generic CC form is further enhanced by Curzon (e.g. *TransWhile bcode ccode base size* - pg. 259, top L para) who also teaches compiler for processing and using a Structured Assembly Language (pg. 258-260). It would have been obvious for one of ordinary skill in the art at the time the invention was made to generate the template as taught by Leeper so that instead of using condition symbol, a cc form as suggested by Curzon represents such condition code because this way the more generic **cc** form can further be translated into a wider range of conditional situations requiring more elaborate operators or symbolology such as taught by known practices of high-language abstraction, thus expanding the useability of condition codes that would otherwise be more limited in Leeper's approach.

As per claim 3, Leeper does not disclose explicitly generating a data structure referencing no arguments, cc, and a branch destination in response to the condition code. But the CC limitation is taught from the teaching of Curzon; hence this teaching would have been obvious in view of the rationale as set forth in claim 2. Leeper, however discloses a condition code without arguments and a destination for branch (see *BRW ELSEBEGINnn* - top para pg. 55 – Note: a 'branch always' is also a condition code wherein no arguments are needed because it is like a if (TRUE) type of assertion); hence this data structure referencing no arguments, a CC, and a destination would also have been obvious by virtue of Leeper's teachings combined with the rationale using Curzon.

As per claim 6, Leeper discloses means:

for recognizing a structured assembly language expression's mnemonics resulting from a logical ANDing of SA_Expr1 and SA_Expr2, wherein each of said SA_Expr1 and said SA_Expr2 is a unit or compound structured assembly language expression (Note: the use of AND operator for logical AND is implicitly disclosed because all high-level languages leading to an assembly language have this operator (e.g. '&&' or AND or inverse-V-Notation) operating on at least 2 arguments to make a single unit of SAL as represented in Leeper's templates);

for setting said branch in each data structure of said SA_Expr1 that is branching to said first branch location to branch to end of said SA_Expr1 (e.g. *THEN_BEGINnn:* – pg. 54, 3rd para; *THEN_BEGIN01:* – pg. 54, last para – Note: from the standpoint in high-level code parsing, branching by taking all the contents of the node to another node, like skipping the entirety of the contents of a subtree in flow graph, i.e. the unexecuted instructions above the adjusted branch destination label, implicitly discloses this limitation).

The limitation for concatenating and preserving order of data structures in said SA_Expr1 and said SA_Expr2 into a single compound structured assembly language expression falls under the known concept of programming languages which teaches a compound logical operation is such that it concatenates orderly simpler logical operations as has been addressed in claim 5; and should be inferred to be disclosed by Leeper; however, this is not explicitly shown by Leeper.

In case Leeper does not teach a compound SAL expression as inferred from high-level programming language, this limitation is taught by Curzon (see pg. 260 R column, 261, R column – Note: code using inverse V for AND operators stands for ANDing in structured language compound expressions). Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide such Logical compound expression thus taught to the template generating by Leeper because this would enhance the logical AND operation so that a number of simple condition can be addressed separately to yield a result by virtue of a compound condition checking as well-known in the art of programming language.

6. Claims 7-8 are rejected under 35 U.S.C. 103(a) as being unpatentable over Leeper et al., "Structured Assembly Language in VAX-11 MACRO", Feb. 1986, Proceedings of the 17th SIGCSE technical symposium on Computer Science education, Vol. 18,m issue 1; and Curzon, Paul, "A Verified Compiler for a Structured Assembly Language", 1992, *International Workshop on Higher Order Logic Theorem Proving and Its Applications*, pp. 253-262; and further in view of Mangelsdorf, USPN: 6,012,836 (hereinafter Mangelsdorf).

As per claim 7, Leeper discloses means for

recognizing a structured assembly language expression's mnemonics requiring a logical ORing of SA_Expr3 and SA_Expr4,wherein each of said SA Expr3 and said SA Expr4 is a unit structured assembly language expression (Note: the use of OR operator for logical OR is implicitly disclosed because all high-level languages to be translated into assembly code have this operator (e.g. '||' or OR or V-Notation) operating on at least 2 arguments to make a single unit of SAL as represented in Leeper's templates); and concatenating and preserving order of data structures in said SA Expr3 and said SA Expr4 into a single compound structured assembly language expression (Note: this limitation would have been implicit or at worst obvious in view of the rationale using Curzon applied to the AND operation from above).

Since Leeper does not explicitly teach compound SAL expression when applying the OR-ing operation thereto, this limitation would also have been obvious in view of the rationale to use the teachings by Curzon (e.g. pg. 260 R column, 261, R column – with the *V* operator standing for OR-ing).

Leeper does not explicitly disclose:

(i) changing said branch location in each of the data structures of SA_Expr3, except the last data structure of SA_Expr3, from said branch location to end of SA_Expr3

(ii) complementing said branch condition in said SA Expr3's last data structure

(iii) changing said branch location in said SA Expr3's last data structure from a branch to said first location to branch to said second location, or from a branch to said second location to branch to said first location.

But based on the understanding from the specifications, these limitations evolve around changing the unit SAL expressions belonging to a compound expression by complementing the condition code operator for each unit SAL expressions and swapping the destination address. And this is reminiscent of an overall expression comprised of '&&' and '|| ' operations wherein modifications of the operators are compensated with the inverting of the partial global result, such result in this instance being represented by a destination address based a TRUE or FALSE state (or partial global result) of the combined sub-expressions evaluation from the && and ||

operations. And by inverting the outcome while complementing the internal operators, the effect is the same as applying a variance of the DeMorgan's theorem, a well-known concept at the time the invention was made. Official notice is taken that the use of DeMorgan's theorem enabling swapping of logical operators to accommodate for environment with restraint in the hardware implementation of specific operator was a known concept at the time the invention was made. For instance, Mangelsdorf, in a method to accommodate for hardware deficiencies, teaches using DeMorgan's approach to eliminate of the NOT operations (col. 16, lines 19-37). In view of the benefits imparted to complementing operators in the inside logical operations and inverting the outside state as known to DeMorgan's theorem and Mangelsdorf's approach, these limitations, i.e. (i) , (ii) and (iii) would have been obvious because applying a variance of the above theorem such as to complementing and inverting the internal logical operators and the outside state, respectively, would yield hardware related benefits favoring a certain instruction set or architecture according to the above Official notice or Mangelsdorf, thus enhancing the assembly language generation for a particular platform machine in which code generation and resources have to be optimized in view of the constraints above.

As per claim 8, this claim corresponds to the limitations of claim 7 for it also include changing/swapping destination location and complementing branch condition code and further includes the use of complementing to substitute for a would-be NOT logical operation as suggested by Mangelsdorf; hence is rejected using the rationale applied to claim 7.

Response to Arguments

7. Applicant's arguments filed 1/18/2005 have been fully considered but they are not persuasive. Following are Examiner's remarks in regard thereto.

Rejection 35 USC §102:

(A) Applicant has submitted that Leeper's cited paragraphs related to IF-THEN and to IF-THEN-ELSE constructs are mutually not related; and that there is no coherency among different concepts (Appl. Rmrks, pg. 9, bottom para). The claim recites means for 'recognizing … expression mnemonics …'; means for 'constructing a data structure referencing …'; means for 'generating a conditional branch …', means for 'generating a first branch'; means for 'generating a first branch' … etc. Apparently, all the claim does is listing of steps for generating elements being recognized from the first recognizing step. The claim amounts to recognizing a form involving 3 elements, and generating a corresponding data structure therefor and further creating each of the code elements represented in said structure. The rejection is based on Examiner's interpretation of what constitutes a structure assembly language mnemonic of the form arg1 cc arg2 and has provided the corresponding mapping for such form. The rejection has presented what constitutes a *data structure* that is referencing said *arg1*, *arg2*, and *cc*; and respectively has mapped from Leeper what is understood from *comparison opcode*, *conditional branch*, *first branch location*, etc. It is purely incidental that the elements used to map those above limitations come from different pages, because it suffices that those elements be generated (whether under an IF-THEN context or a IF-THEN-ELSE context), to met the claim requirement. Indeed, the claim does not compel that one conditional opcode has to come from or map exactly a specific cc, or arg1 or arg2; and that by cc it can only be one specific cc, based on the reciting of loosely established limitation of the likes of 'in response to elements of said data structure'. So long as these elements are generated as a result of recognizing (Note: *recognizing* does not compel a required one-to-one mapping either) a *arg1 cc arg2* expresssion which is viewed as being

translated into Leeper's template expression, as set forth in the rejection, the claim limitations are met. For the sake of arguments, it can be explained that the corresponding structure language templates (data structure as a result of a particular mnemonic being recognized) in page 54, 55 or 56 are to be mapped with the code constructs of pg. 54, 55, or 56 respectively; as follows: (structured template) *CMPx* – pg. 54 → (assembly language) CMPC3 – pg. 54, 3rd para;
CMPx – pg. 55, 1st para → CMPL – pg. 55 middle para;
CMPx – pg. 56, 6th para → CMPC3 – pg. 57

The claim fails to provide a clear mapping relating elements of the mnemonic to each elements of the data structure, or from such data structure to the code elements derived therefrom; and by just reciting for instance 'referencing elements of said arg1, arg2, and cc', the claim is insufficiently clear as to force why a particular mapping scheme or order has to be respected as being inferred via Applicant's arguments. Because the arguments seem to demand that a context mapping be shown for coherency, it is noted that in view of the relative broadness of the claim language the prospect of mapping elements among themselves is not clear or strongly expected, let alone mapping them into a context (e.g. IF-THEN, IF-THEN-ELSE, LOOP). Besides, the context issue has been considered moot in light of the incidental citing the above section has shown. The arguments are hence non persuasive.

(B) Applicant has submitted that the IF-THEN construct as shown on Leeper's page 54 is not the data structure as claimed (Appl. Rmrks, pg. 10, top). Once more, the claimed feature is a data structure; and due to the very broad nature of the term there is not an undeniable requirement that the recited 'data structure' should enforce a particular connotation that is to be unique. The IF-THEN template therefore reads on some construct that reference the elements of the recognized mnemonics containing arg1, cc, and arg2; and this is set forth in the rejection, notwithstanding the lack of mapping specificity as mentioned above in section A.

(C) Applicant has submitted that the cited portions shown on Leeper's pages 54-55 are not teaching a comparison opcode in response to arg1, cc, and arg2, branch destination as claimed (Appl. Rmrks, pg. 10, 2nd para). By merely stating that the references does not teach or suggest a claimed invention without pointing specifically where the references as cited fail to meet what Applicant perceives as his invention, the Applicant does not provide sufficient prime facie case of rebut against the rejection; and the arguments amount to mere allegations.

The rejection will stand as set forth above.

Conclusion

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 703-872-9306 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
May 23, 2005

[illegible]

TODD INGBERG
PRIMARY EXAMINER